УДК 519.6

СИСТЕМА УПРАВЛЕНИЯ ЭКСПЕРИМЕНТОМ И ОБРАБОТКИ ДАННЫХ, ПОЛУЧЕННЫХ МЕТОДАМИ ЦИФРОВОЙ ТРАССЕРНОЙ ВИЗУАЛИЗАЦИИ (ACTUALFLOW)

Е. К. Ахметбеков[1], А. В. Бильский[2], Ю. А. Ложкин[1], Д. М. Маркович[2], М. П. Токарев[2], А. Н. Тюрюшкин[3]

В статье рассматривается архитектура программного обеспечения для измерительного комплекса на основе цифровой трассерной визуализации. Представлены подходы к организации хранения данных, расширению системы за счет новых расчетных процедур, визуализации данных. Обсуждается работа с универсальной библиотекой устройств.

Ключевые слова: измерительные комплексаы, визуализация, программное обеспечение, базы данных, цифровые камеры, вычислительные эксперименты.

1. Введение. Программное обеспечение ActualFlow было разработано как часть измерительного комплекса "ПОЛИС", реализующего методы PIV, PTV, LIF, PLIF. Указанные методы возникли сравнительно недавно и получили широкое распространение благодаря новым возможностям, которые они предоставляют. Преимуществами методов являются невозмущающий (бесконтактный) способ измерения, возможность получения принципиально новой информации об объекте.

Метод Particle Image Velocimetry (PIV) позволяет измерять поля скорости в выбранном сечении потока жидкости или газа. В основе метода лежит измерение перемещений взвешенных в потоке частиц, освещенных лазерным импульсом и регистрируемых цифровыми камерами. Модификацией метода является метод Particle Tracking Velocimetry (PTV), отличающийся подходами к расчету поля скорости. Метод Laser Induced Fluorescence (LIF) применяется для измерения полей скорости газовой и жидкой фаз в пузырьковых потоках. Метод Planar Laser Induced Fluorescence (PLIF) позволяет регистрировать мгновенные поля температуры (или концентрации) в потоках жидкости и газа. В основе измерений температуры лежит зависимость интенсивности флюоресценции некоторых красителей от температуры.

Программное обеспечение ActualFlow (ПО) позволяет проводить эксперимент, т.е. управлять работой нескольких аппаратных устройств (лазер, камера, синхронизирующий процессор и др.) и сохранять полученные с них данные, а также обрабатывать данные и отображать результат обработки пользователю различными способами.

Основная область применения измерительных комплексов — это научно-исследовательские и опытно-конструкторские работы. Задачи, которые будут решаться пользователем, зачастую не известны заранее. Поэтому требования, предъявляемые к ПО, предполагают наличие возможности для конечного пользователя самостоятельного расширения функциональности, интегрирования в систему собственных расчетных процедур (алгоритмов обработки данных) и модулей управления входящим в комплект оборудованием.

Все описанные выше методы измерений в качестве исходных данных для обработки используют изображения с цифровой камеры. Большой объем получаемых данных и ресурсоемкость расчетных процедур не позволяют проводить обработку в режиме реального времени. Размер баз данных исчисляется десятками и сотнями гигабайт информации, в связи с чем актуальна задача оптимизации скорости, надежности и удобства работы с большими объемами данных.

В мире существует несколько аналогов программного обеспечения, описанного в данной работе. Это решения, сопровождающие измерительные комплексы компаний Dantec Dynamics (Дания), TSI (США), La Vision (Германия) и др. Все они обладают примерно одинаковой функциональностью, содержат разнообразные процедуры обработки и отображения данных, а также поддерживают некоторое количество устройств из собственной линейки оборудования. Однако большинство программных продуктов не позволяют пользователю интегрировать в систему свои устройства и расчетные процедуры, и для решения нестандартных задач необходимо пользоваться дополнительным инструментарием.

[1] Новосибирский государственный университет, физический факультет, ул. Пирогова, д. 2, 630090, г. Новосибирск; e-mail: akhmetbekov@mail.ru, lozhkin@gmail.com

[2] Институт теплофизики СО РАН, просп. Академика Лаврентьева, д. 1, 630090, г. Новосибирск; e-mail: bilsky@itp.nsc.ru, dmark@itp.nsc.ru, mtokarev@itp.nsc.ru

[3] ООО Промис Плюс, ул. Институтская, д. 6, 630090, г. Новосибирск; e-mail: dduuhhaa@gmail.com

С точки зрения информационных технологий ActualFlow представляет собой сложный расчетный комплекс с распределенной модульной архитектурой, основанной на COM-технологии межсервисного и межкомпонентного взаимодействия. В качестве основных единиц декомпозиции продукта можно выделить следующие:

— блок хранения данных;

— пользовательский интерфейс с возможностями импорта/экспорта и отображения данных;

— блок управления экспериментом;

— блок обработки данных, включающий набор готовых расчетных процедур (алгоритмов) и модуль управления обработкой.

Алгоритмы обработки данных разбиты по группам (методам) и могут комплектовать программный продукт в зависимости от задач потребителя. Модуль интеграции оборудования и модуль управления обработкой предоставляются пользователю с открытым интерфейсом (API), документацией и с примерами реализации собственных расширений, что делает систему гибкой и расширяемой.

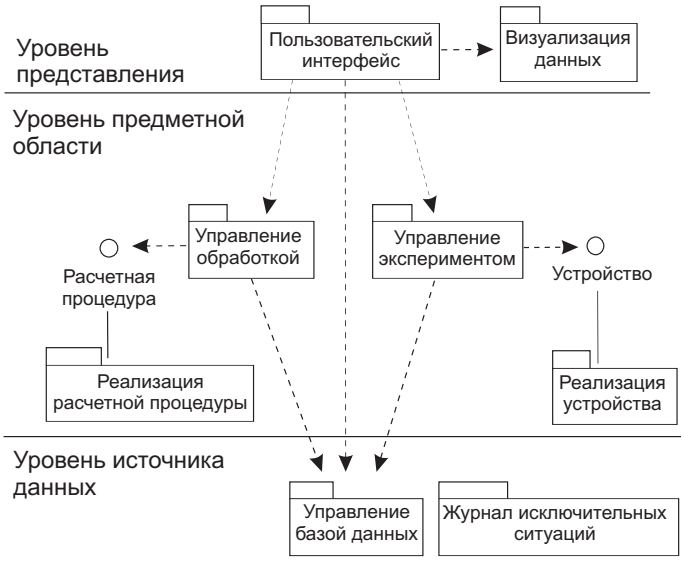

Рис. 1. Послойное представление системы

В статье представлено описание ПО ActualFlow: архитектура, взаимодействие программных модулей между собой и с пользователем, способы организации хранения и обмена данными. Статья может быть полезна разработчикам средств автоматизации контрольно-измерительного оборудования, характеризующегося большими объемами данных и разнообразием интегрированных аппаратных модулей и/или расчетных процедур.

2. Архитектура и общая характеристика программного продукта ActualFlow. Разбиение системы на модули соответствует классическому делению программного обеспечения на три основных логических слоя [5]: уровень представления, уровень предметной области и уровень источника данных (рис. 1). Деление на слои позволяет уменьшить зависимости между логическими уровнями системы, увеличить локальность изменений в программе, добавить возможность альтернативной реализации базового слоя (например, уровня источника данных), улучшить понимание функционирования программы разработчиками. К недостаткам такого деления можно отнести лишь незначительное снижение производительности системы за счет необходимости организации межслойного взаимодействия и конвертации/преобразования данных. На данном этапе все модули программной системы располагаются физически в пределах одной машины, однако могут быть распределены на несколько машин. Нижний уровень источника данных представлен модулем управления данными и модулем, реализующим журнал исключительных ситуаций. Модуль управления данными изолирует конкретную организацию базы данных от остальной части системы и предоставляет верхним слоям системы интерфейс для оперирования данными в виде объектов посредством установки их свойств и посылки различных сообщений. Журнал исключительных ситуаций используется всеми модулями для трассировки событий, происходящих в системе.

Уровень предметной области включает всю бизнес-логику программного обеспечения: блок обработки данных и блок управления экспериментом. Этот уровень открыт для расширения за счет интеграции новых расчетных процедур и устройств. Замыкает послойное разбиение верхний уровень представления, который образован интерфейсом пользователя и компонентом визуализации данных.

Для межсессионного сохранения настроек пользовательского интерфейса (палитра, параметры визуализации данных), цепочек обработки и метаинформации узлов базы данных применяются файлы в формате XML. Некоторые дополнительные настройки сохраняются в системном реестре Windows.

Программное обеспечение реализовано на объектно-ориентированном языке программирования C++ для работы под управлением операционных систем семейства Windows 2000/XP. Минимальные требования к аппаратному обеспечению, при условии использования ПО для управления экспериментом, следующие: процессор не ниже Pentium III, 256 Мб оперативной памяти, USB и COM порты.

Ниже будут рассмотрены некоторые особенности реализации отдельных модулей программного обеспечения.

3. Хранение данных. База данных (БД) должна хранить информацию о проводимых экспери-

ментах, которая включает в себя информацию об устройствах и параметрах их работы. Структура БД древовидная, отдельный элемент которой называется узлом. В настоящее время в БД выделены следующие типы узлов: база данных, эксперимент, папка (организующий элемент для системы экспериментов внутри БД), изображение и элемент данных. Каждый узел эксперимента может содержать группу изображений и группу внешних импортируемых данных. В свою очередь, каждый элемент внутри узла эксперимента содержит данные, которые были получены путем применения расчетных процедур. Таким образом, связями потомок–родитель внутри узла эксперимента кодируется информация о том, каким методом и из каких данных был получен текущий узел.

Физическая организация данных представляет собой спроектированную специально под данную задачу файловую БД, состоящую из файла описания иерархии узлов (файл структуры) и системы файлов с вложенными подкаталогами. Такая организация данных продиктована особенностью проведения измерений и оптимизацией скорости чтения иерархии узлов.

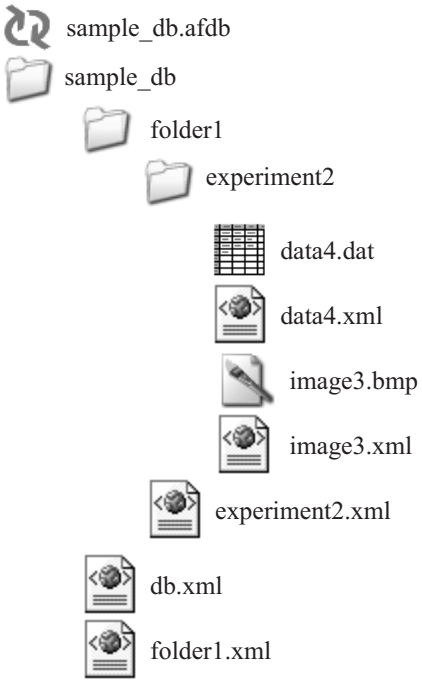

Рис. 2. Расположение элементов БД в файловой системе

Система из файлов и каталогов организована следующим образом. Информация о конкретном узле разделена на описание и данные узла. Описание узла, или метаинформация, хранится в XML файле. Каждый файл метаинформации содержит версию формата, общую информацию для всех узлов и теги специфической информации для каждого типа узла. К общей информации относятся, например, время создания узла, текстовое описание, название, уникальный идентификатор, тип узла и др. Одно из важных достоинств представления описания узлов в XML формате состоит в нисходящей и восходящей совместимости версий форматов представления данных.

Формат файла с данными зависит от конкретного типа узла. Растровые изображения, соответствующие узлу изображения в базе, могут храниться в одном из общепринятых форматов: bmp, tiff, jpg, png. Данные, соответствующие двумерным полям и одномерным распределениям значений, хранятся в файлах в бинарном представлении. Такие данные представляют собой набор значений, находящихся на регулярной либо нерегулярной сетке. Например, для двумерного двухкомпонентного поля скорости набор значений состоит из координат узлов сетки (x, y) и двух компонент векторов скорости $(\boldsymbol{u}, \boldsymbol{v})$ в этих узлах.

Расположение файлов базы на диске выглядит, как показано на рис. 2. Для таких организующих элементов как папка (на рисунке — “folder1”) и эксперимент (на рисунке — “experiment2”) их непосредственные потомки находятся в каталоге с соответствующим названием. В предложенной БД реализована двойная косвенность положения конечных файлов с данными относительно файла описания структуры базы и относительно XML-файла описания узла. Использование относительных путей позволяет переносить БД с диска на диск без ее модификаций. Относительные ссылки на данные из файлов метаинформации дают возможность переноса данных из главного каталога базы в любой другой каталог или на другой диск, если были исчерпаны ресурсы текущего диска.

По результатам тестов предложенная модель организации данных дает существенный (в два с половиной раза) выигрыш в скорости операций с данными по сравнению с реализацией на СУБД MSSQL.

4. Обработка данных. Обработка данных в рассматриваемой программной системе является одной из основных прикладных функций. Функции обработки реализованы в модуле управления обработкой и библиотеке расчетных процедур (алгоритмов). В таблице представлены процедуры обработки данных, реализованные в настоящий момент.

Модуль управления обработкой создает план обработки на основе выбранной пользователем группы входных данных и последовательности расчетных процедур, которую необходимо применить к этим данным. Здесь же ведется трассировка основных операций в журнал обработки с фиксированием времени начала и продолжительности работы каждого алгоритма в цепочке, длительности операций ввода/вывода. Одна из задач данного модуля состоит в двунаправленном преобразовании данных, получаемых от алгоритма во внутренний формат данных, с которым работает источник данных, рассмотренный в предыдущем разделе. Использование собственного формата данных для алгоритмов связано как с вопро-

Базовые алгоритмы:	— стандартный и адаптивные кросскорреляционные алгоритмы расчета поля скорости методом PIV/LIF [4]; — процедуры отсева ошибочных векторов по критериям локальной гладкости векторного поля, по направлению и модулю скорости, по критерию сигнал/шум; — расчет поля скорости методом PTV [2]); — процедуры калибровки и расчета полей температуры методом PLIF [3].
Дополнительные алгоритмы:	— процедура расчета статистических моментов измеряемых величин (до 4-го порядка включительно); — метод статистической фильтрации ансамблей полей скорости на основе анализа гистограмм распределения скорости [1]; — алгоритмы интерполяции и расчета пространственных производных; — алгоритмы калибровки и расчета трехкомпонентных полей скорости; — расчет пространственных спектров и пространственных корреляций.
Утилиты:	— построение профиля на основе двумерного поля данных; — процедуры маскирования областей на изображениях и полях данных; — арифметические операции и расчет статистики по группе изображений.

сом восходящей совместимости, так и с необходимостью изолировать процедуры расчета от внутреннего представления данных для возможности использования их отдельно от системы. Интерфейс расчетной процедуры был разработан таким образом, чтобы использовать только COM-специфицированные механизмы передачи данных (без неявного приведения типов по указателю), что позволяет работать с алгоритмами непосредственно из клиентов, созданных на различных языках программирования — Visual Basic, C++, Delphi и др.

Расчетные процедуры группируются в DLL-библиотеки. Библиотека содержит COM-классы расчетных процедур и предоставляет доступ к их описанию. COM-класс алгоритма реализует интерфейс, включающий в себя следующие элементы:

1) передача списка входных параметров и их значений;

2) передача входных/выходных данных.

Это базовые составляющие интерфейса. Для обеспечения удобства работы с системой в него были добавлены:

3) доступ к значениям параметров по умолчанию;

4) отображение диалогового окна для задания входных параметров алгоритма;

5) хранение графической иконы для визуальной идентификации данных, полученных соответствующим алгоритмом.

При передаче большого объема данных на вход алгоритма часто невозможно передать данные за один раз, вследствие ограниченности размера оперативной памяти. Для этого организован запрос дополнительной порции данных через механизм Connection point, который также используется для информирования о процессе расчета.

Таким образом, блок обработки предоставляет набор процедур обработки, реализующих унифицированный интерфейс. Программный продукт содержит обширную библиотеку готовых алгоритмов, а также позволяет пользователю интегрировать в систему собственные алгоритмы, что является одним из существенных преимуществ перед другими системами.

5. Сбор данных. Одним из этапов получения данных об исследуемом объекте является проведение эксперимента. В задачи блока управления экспериментом входит:

— управление периферийными устройствами;

— установка параметров эксперимента и отдельных устройств;

— синхронизация устройств во время проведения эксперимента;
— опрос состояния устройств;
— передача полученных данных основному модулю.

Кроме того, со стороны других модулей необходима функциональность для приема данных, интерпретации их в соответствии с параметрами эксперимента, сохранения в базе данных и визуализация в режиме реального времени.

Модуль управления экспериментом реализован в виде ATL COM библиотеки. Он производит инициализацию устройств, которые были выбраны при создании эксперимента, и предоставляет диалоговое окно с настройками этих устройств и элементами управления экспериментом. Каждое устройство, которое имеет параметры управления, предоставляет окно настроек в виде закладки в основном диалоговом окне. В эксперименте может участвовать мастер-устройство, которое задает сценарий проведения эксперимента. В типичной конфигурации таким устройством является синхронизирующий процессор. Во время эксперимента все устройства, являющиеся источниками данных, опрашиваются на наличие данных и при их возникновении они передаются основному модулю через COM интерфейс обратного вызова — Connection Point. После запуска сбор данных протекает в соответствии с заданным сценарием и останавливается по завершении программы сбора, но может быть остановлен пользователем в произвольный момент времени.

Аппаратная часть измерительного комплекса “ПОЛИС” (см. рис. 3) включает цифровые камеры на базе CCD матрицы. Камеры применяются для регистрации мгновенных изображений потока. Управление камерой и передача данных осуществляется через PCI плату (Frame Grabber). Для освещения измерительной области в потоке используется твердотельный импульсный Nd:YAG лазер с длиной волны 532 нм, луч которого при помощи специализированного оптического модуля разворачивается в плоскость регулируемой толщины.

Рис. 3. Схема взаимодействия элементов системы “ПОЛИС”

Для синхронизации работы аппаратных устройств в эксперименте участвует синхронизирующий процессор. Процессор задает количество и частоту измерений, выдает управляющие TTL сигналы, запускает систему от внешнего события и т.д. Синхронизирующий процессор представляет собой программируемый таймер с несколькими выходными линиями стандартного TTL сигнала. Программирование устройства осуществляется через последовательный порт RS 232 (COM port), с помощью которого по определенному протоколу задаются значения внутренних регистров таймера, производится запуск и остановка системы. Данный прибор работает как в полностью автономном режиме, так и в режиме внешнего запуска, который необходим для проведения измерений, привязанных к внешнему событию. Выделение функции синхронизации работы устройств в отдельное устройство (процессор) обусловлено тем, что временна́я точность выполняемых действий должна быть не хуже 10 нсек, а операционная система Windows не гарантирует такой точности.

Реализованная система предполагает расширение не только за счет поддержки все большего количества камер и лазеров, но и за счет появления принципиально новых устройств. По этой причине архитектура блока управления экспериментом построена на использовании минимального количества допущений об устройстве, что, с другой стороны, усложняет логику их взаимодействия. Блок управления каждого из устройств является отдельной COM библиотекой. В задачи библиотеки устройства входит предоставление графического окна в виде закладки с параметрами, взаимодействие с остальными устройствами и передача данных, если таковые имеются.

6. Визуализация данных. Функциональность системы, связанная с представлением данных пользователю, объединена в модуль визуализации. При разработке модуля визуализации решались следующие задачи: отображение всех видов данных, с которыми работает система, и обеспечение наиболее продуктивного способа работы с отображаемыми данными для пользователя. Кроме того, архитектура модуля должна быть достаточно гибкой и легко расширяемой, так как система постоянно развивается и введение

новых видов данных зачастую требует введения новых способов визуализации.

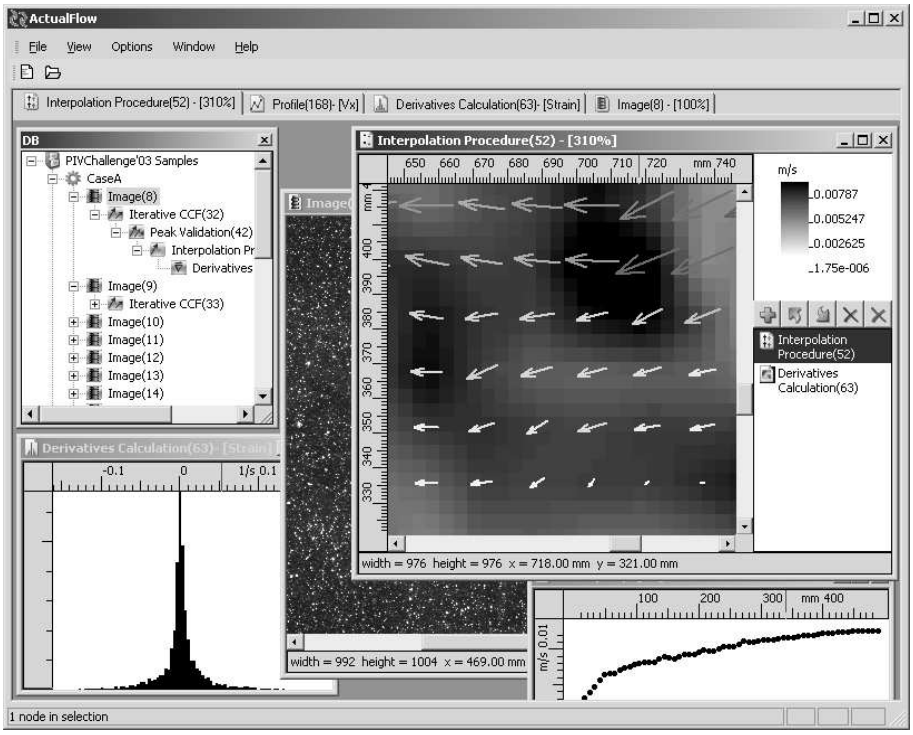

Рис. 4. Способы визуализации данных: растровые изображения, двумерное распределение, гистограмма

Исходя из решаемых большинством пользователей задач, было выделено три базовых способа отображения данных (см. рис. 4):

— поле (двумерное распределение);

— график (одномерное распределение);

— гистограмма, которая может быть построена как по полю, так и по графику.

Эти три способа отображения данных существуют в системе независимо; для их реализации использован общий архитектурный шаблон, включающий в себя следующие абстракции (см. рис. 5):

— окно;

— фабрика визуализируемых объектов;

— визуализируемый объект;

— диалог настройки параметров визуализации.

Окно предоставляет пользователю средства для анализа данных и управляет объектами визуализации. Возможно одновременное отображение нескольких объектов в одном окне. Операции создания объекта визуализации узла базы данных согласно типу узла делегированы фабрике объектов. Отрисовка данных выполняется объектом визуализации, при этом оконный класс управляет масштабированием и взаимным расположением объектов на области построения. Каждый объект визуализации предоставляет диалог настройки параметров отображения. Для доступа к данным используются обработчики форматов файлов.

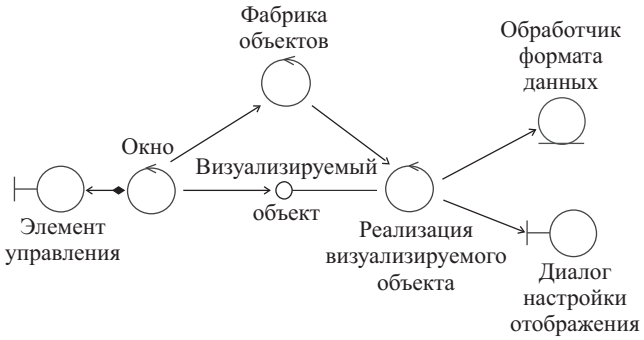

Рис. 5. Архитектурный шаблон, использованный при реализации модуля визуализации

Рассмотрим реализацию этого проектного решения на примере отображения полей данных (активное окно на рис. 4). Двумерное поле — это основной результат измерений при помощи описанных выше методик. К полям данных относятся изображения, поля скорости (PIV, PTV), поля температуры и др. Выделяются следующие виды визуализируемых объектов: векторное поле, скалярное поле, маска, растровое изображение. Окно построения полей содержит

оси координат, легенду, палитру (значение в точке скалярного поля и длина вектора кодируются цветом) и список открытых в окне объектов. Предоставляются средства для масштабирования изображения в области построения, вызова диалогов настройки параметров отображения, добавления и удаления объектов визуализации из списка объектов. Кроме того, возможно наложение различных способов отображения данных. Например, векторное поле поверх скалярного (см. рис. 4).

В случае появления новых видов полей достаточно добавить еще один объект визуализации и конструирующий его код. Однако можно воспользоваться существующим типом данных, например "поле с регулярной сеткой", и тогда данные будут отображаться в виде скалярного поля без внесения изменений в модуль визуализации.

Для упрощения и унификации работы с данными узла была выделена абстракция обработчика формата (итератора) данных, позволяющего работать со всеми полями из файла, например, со значениями одной из компонент скорости. Итераторы создаются по запросу объектов визуализации отдельной фабрикой объектов, позволяющей получить список полей узла. Такой подход существенно упрощает обработку форматов данных, обеспечивает прозрачный для пользователя доступ к данным, позволяет вводить искусственные (отсутствующие в файле, но полезные для анализа) поля данных.

7. Заключение. В статье представлена общая архитектура и особенности реализации программной системы ActualFlow, которая позволяет проводить эксперимент и сохранять полученные данные, а также обрабатывать данные и отображать результат обработки пользователю.

Система разбита на три логических слоя: уровень представления (пользовательский интерфейс), уровень предметной области (блок управления экспериментом и блок обработки данных) и уровень источника данных (блок хранения данных). Логическая структура базы данных — дерево, в котором связями потомок–родитель кодируется информация о том, каким методом и из каких данных был получен текущий узел. Физически база данных представляет собой файл описания иерархии узлов и систему файлов с вложенными подкаталогами. Блок обработки включает в себя модуль управления обработкой и обширную библиотеку готовых процедур обработки данных, открытую для расширения. Блок управления экспериментом управляет внешними устройствами и сохраняет полученные данные в базе. Возможна интеграция управляющих модулей как для новых камер и лазеров, так и для принципиально новых устройств. Пользовательский интерфейс предоставляет средства для наглядного управления всеми частями системы и визуализации данных. Данные отображаются в виде полей, графиков и гистограмм.

Представлен один из вариантов эффективного и наглядного хранения больших объемов данных и их обработки. Описан подход к организации расширения системы конечным пользователем за счет интеграции новых расчетных процедур и управляемого оборудования, что существенно расширяет область применения продукта.

В настоящее время ActualFlow используется в экспериментах по исследованию потоков жидкости и газа в нескольких научных коллективах.

СПИСОК ЛИТЕРАТУРЫ

1. *Heinz O.M., Ilyushin B.B., Markovich D.M.* Application of a PDF's method for the statistical processing of experimental data // Int. Journal of Heat and Fluid Flow. 2004. **25**. 846–874.
2. *Baek S.J., Lee S.J.* A new two-frame particle tracking algorithm using match probability // Experiments in Fluids. 1996. **22**. 23–32.
3. *Coolen M.C.J., Kieft R.N., Rindt C.C.M., Steenhoven A.A.* Application of 2-D LIF temperature measurements in water using a Nd:YAG laser // Experiments in Fluids. 1999. **27**. 420–426.
4. *Raffel M., Willert C., Kompenhans J.* Particle image velocimetry. A practical guide. Berlin: Springer, 1998.
5. *Фаулер М.* Архитектура корпоративных программных приложений. Киев: Вильямс, 2004.

Поступила в редакцию
20.06.2006